

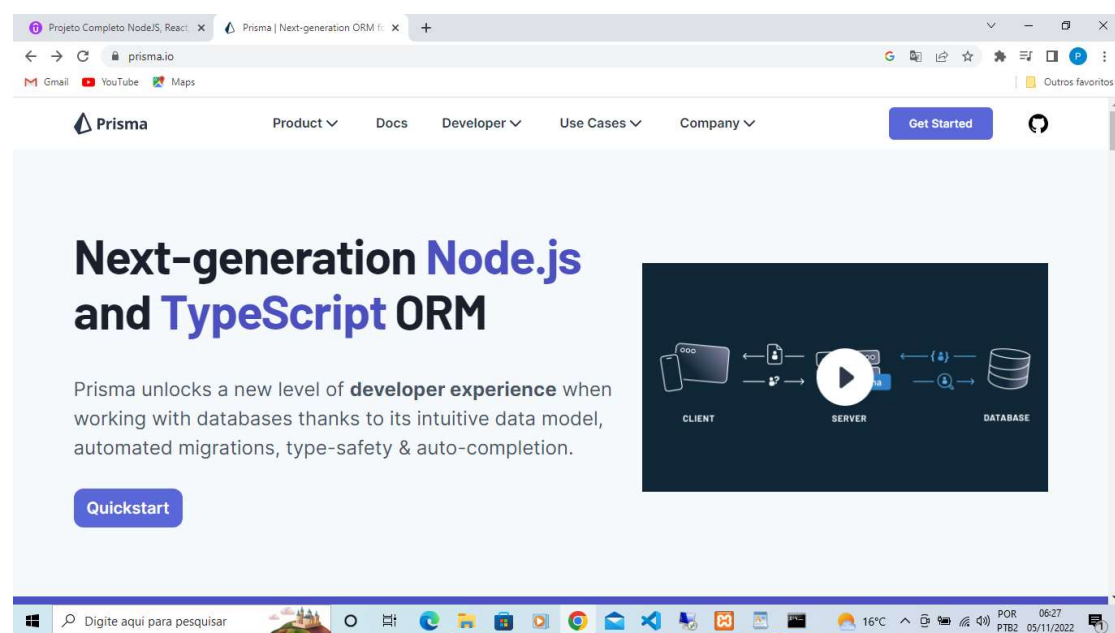
Fatec Praia Grande

Paulo R. T. Cândido

NodeJS - parte 2

Construção de CRUD para produtos

1) Instalação do PRISMA, ORM para interação com banco de dados.



```
yarn add prisma
```

```
yarn add @prisma/client
```

```
npx prisma init
```

2) Configurar o BD

Obs.: Criar no MySql o banco de dados MEUDB

```

prisma > schema.prisma
1 // This is your Prisma schema file,
2 // learn more about it in the docs: https://pris.ly/d/prisma-schema
3
4 generator client {
5   provider = "prisma-client-js"
6 }
7
8 datasource db {
9   provider = "mysql"
10  url      = env("DATABASE_URL")
11 }

```

```

1 # Environment variables declared in this file are automatically made available to Prisma.
2 # See the documentation for more detail: https://pris.ly/d/prisma-schema#accessing-environment-variables-from-the-schema
3
4 # Prisma supports the native connection string format for PostgreSQL, MySQL, SQLite, SQL Server, MongoDB and CockroachDB.
5 # See the documentation for all the connection string options: https://pris.ly/d/connection-strings
6
7 DATABASE_URL="mysql://root:@localhost:3306/meubd?schema=public"

```

3) Definir o Model no arquivo schema.prisma

// This is your Prisma schema file,
 // learn more about it in the docs: <https://pris.ly/d/prisma-schema>

```

generator client {
  provider = "prisma-client-js"
}

datasource db {
  provider = "mysql"
  url      = env("DATABASE_URL")
}

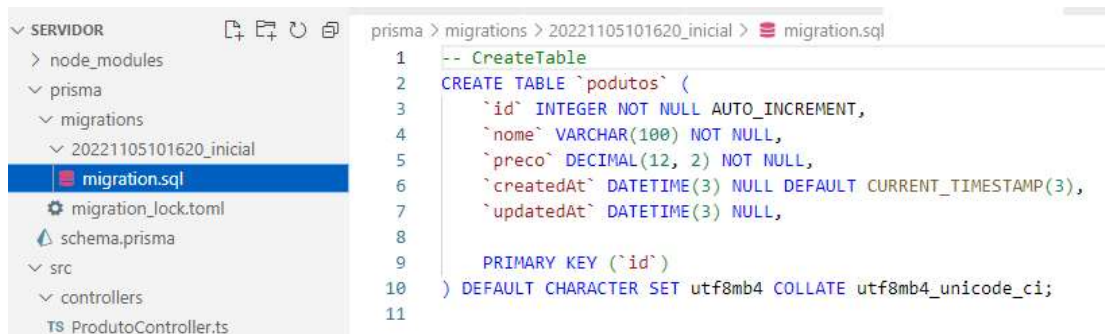
model Produto {
  id Int @id @default(autoincrement())
  nome String @db.VarChar(100)
  preco Decimal @db.Decimal(12,2)
  createdAt DateTime? @default(now())
  updatedAt DateTime? @updatedAt

  @@map("produtos")
}

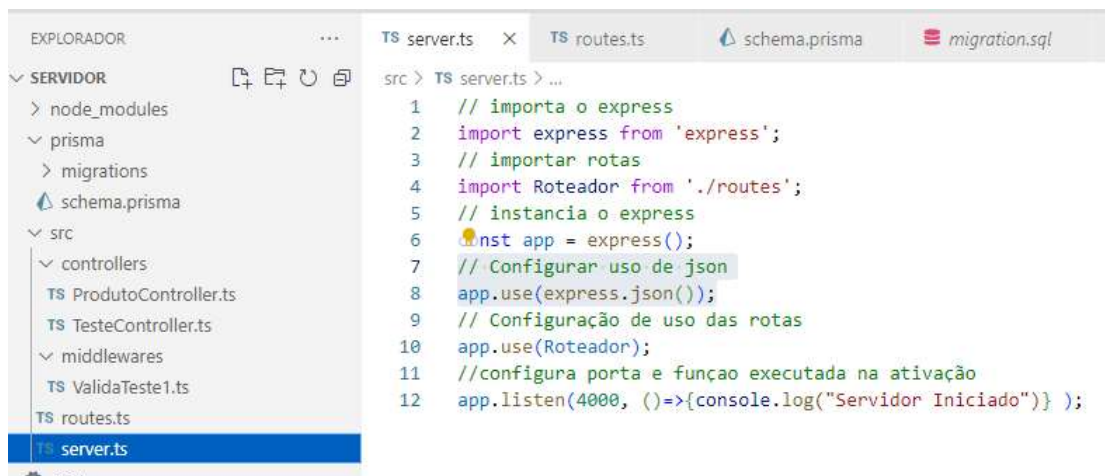
```

4) Criar e executa migração que irá criar no BD a tabela produtos (@@map("produtos"))

yarn prisma migrate dev --name inicial



5) Incluir suporte a Json do express



6) Criar o controller para implementar o CRUD

```

import {Request, Response} from 'express';
import {PrismaClient} from '@prisma/client';

```

```

class ProdutoController {
  async index(req:Request,res:Response){
    const prisma = new PrismaClient();
    const produtos = await prisma.produto.findMany(); // recupera todos os produto
    res.status(200).json(produtos);
  }

  async show(req:Request,res:Response){
    const prisma = new PrismaClient();
    const produto = await prisma.produto.findUnique( // busca produto conforme where
      {
        where:{id: Number(req.params.id)},
        select:{id:true,nome:true,preco:true} // quais dados se quer no resultado
      }
    );
    res.status(200).json(produto);
  }

  async store(req:Request,res:Response){
    const prisma = new PrismaClient();
    //obtem json vindo do cliente
    const dados = req.body;
    //console.log(dados);
    const novoPoduto = await prisma.produto.create(

```

```

        {
          data: dados,
          select: {
            id:true,
            nome:true,
            preco:true
          }
        }
      );
      res.status(200).json(novoPoduto);
    }

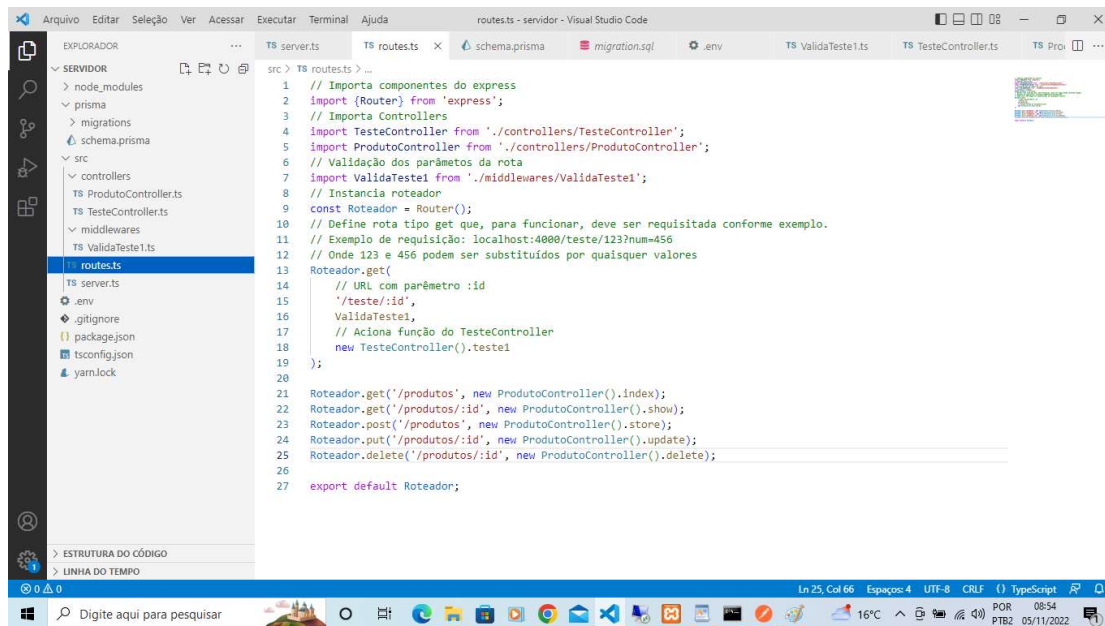
    async update(req:Request,res:Response){
      const prisma = new PrismaClient();
      const produtoAlterado = await prisma.produto.update(
        {
          where: {id: Number(req.params.id) },
          data: req.body,
          select: {
            id:true,
            nome:true,
            preco:true
          }
        }
      );
      res.status(200).json(produtoAlterado);
    }

    async delete(req:Request,res:Response){
      const prisma = new PrismaClient();
      await prisma.produto.delete(
        {
          where: {id: Number(req.params.id) }
        }
      );
      res.status(200).json({excluido: true});
    }
  }
}

export default ProdutoController

```

7) Configurar rotas



```
1 // Importa componentes do express
2 import { Router } from 'express';
3 // Importa Controllers
4 import TesteController from './controllers/TesteController';
5 import ProdutoController from './controllers/ProdutoController';
6 // Validação dos parâmetros da rota
7 import ValidadeTeste1 from './middlewares/ValidadeTeste1';
8 // Instancia roteador
9 const Roteador = Router();
10 // Define rota tipo get que, para funcionar, deve ser requisitada conforme exemplo.
11 // Exemplo de requisição: localhost:4000/teste/123?num=456
12 // Onde 123 e 456 podem ser substituídos por quaisquer valores
13 Roteador.get(
14   // URL com parâmetro :id
15   '/teste/:id',
16   ValidadeTeste1,
17   // Ação função do TesteController
18   new TesteController().teste1
19 );
20
21 Roteador.get('/produtos', new ProdutoController().index);
22 Roteador.get('/produtos/:id', new ProdutoController().show);
23 Roteador.post('/produtos', new ProdutoController().store);
24 Roteador.put('/produtos/:id', new ProdutoController().update);
25 Roteador.delete('/produtos/:id', new ProdutoController().delete);
26
27 export default Roteador;
```

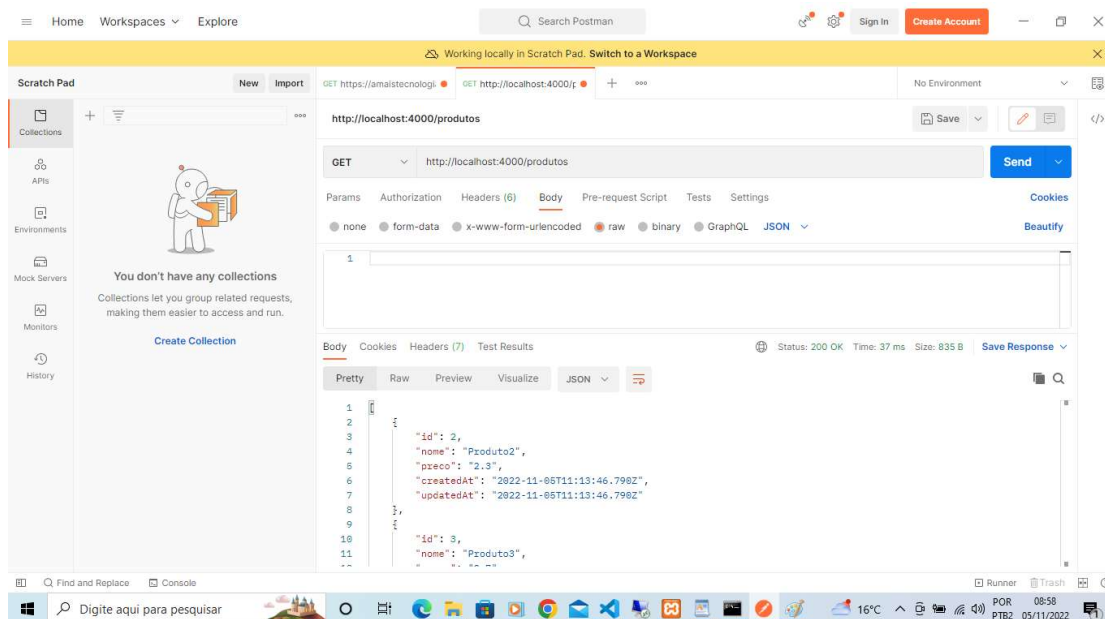
Roteador.get('/produtos', new ProdutoController().index);
Roteador.get('/produtos/:id', new ProdutoController().show);
Roteador.post('/produtos', new ProdutoController().store);
Roteador.put('/produtos/:id', new ProdutoController().update);
Roteador.delete('/produtos/:id', new ProdutoController().delete);

8) Startar a aplicação

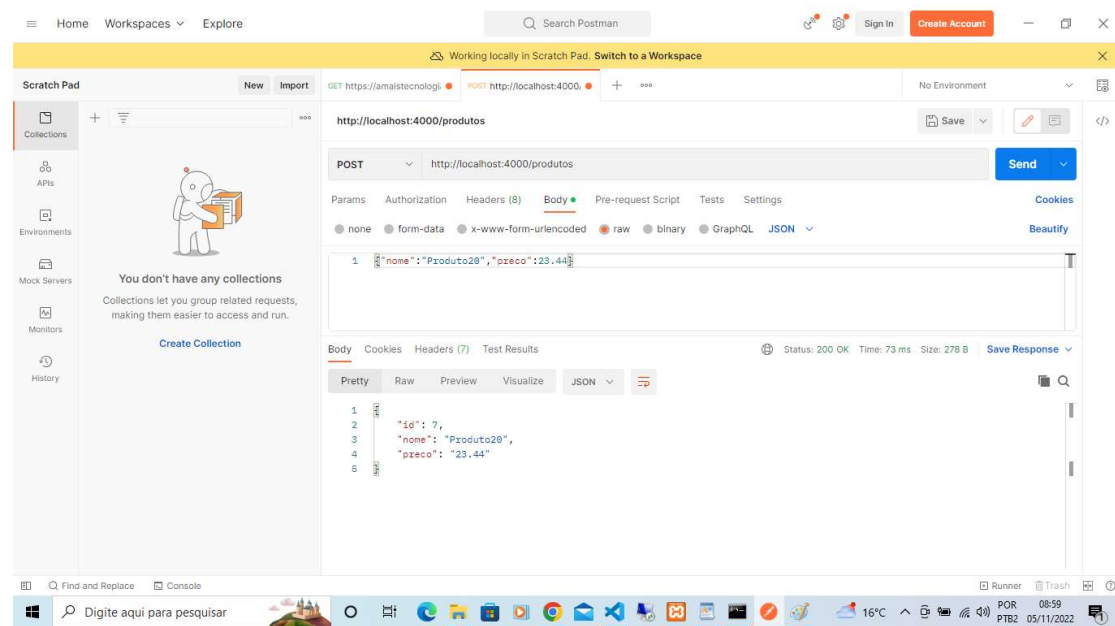
yarn dev

9) Testar com o Postman

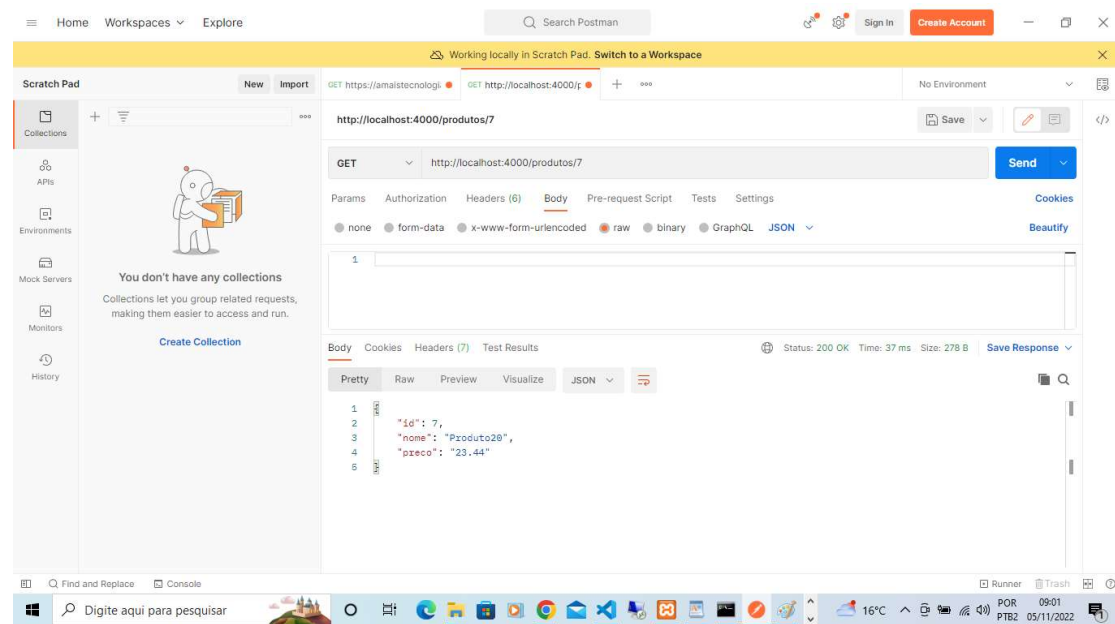
Consultar todos



Cadastra novo



Consultar um produto



Atualizar um produto

Home Workspaces Explore Search Postman Sign In Create Account

Working locally in Scratch Pad. Switch to a Workspace

Scratch Pad New Import GET https://lamaitecnologi PUT http://localhost:4000/p

collections

You don't have any collections
Collections let you group related requests, making them easier to access and run.
Create Collection

APIs
Environments
Mock Servers
Monitors
History

http://localhost:4000/produtos/7

PUT http://localhost:4000/produtos/7 Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 { "nome": "Produto28Alterado", "preco": 37.93 }
```

Body Cookies Headers (7) Test Results Status: 200 OK Time: 38 ms Size: 286 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 7,
3   "nome": "Produto28Alterado",
4   "preco": 37.93
5 }
```

Find and Replace Console Runner Trash

16°C 09:02 05/11/2022

Excluir produto

Home Workspaces Explore Search Postman Sign In Create Account

Working locally in Scratch Pad. Switch to a Workspace

Scratch Pad New Import GET https://lamaitecnologi DEL http://localhost:4000/p

collections

You don't have any collections
Collections let you group related requests, making them easier to access and run.
Create Collection

APIs
Environments
Mock Servers
Monitors
History

http://localhost:4000/produtos/7

DELETE http://localhost:4000/produtos/7 Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1
```

Body Cookies Headers (7) Test Results Status: 200 OK Time: 34 ms Size: 252 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "excluido": true
3 }
```

Find and Replace Console Runner Trash

16°C 09:03 05/11/2022

